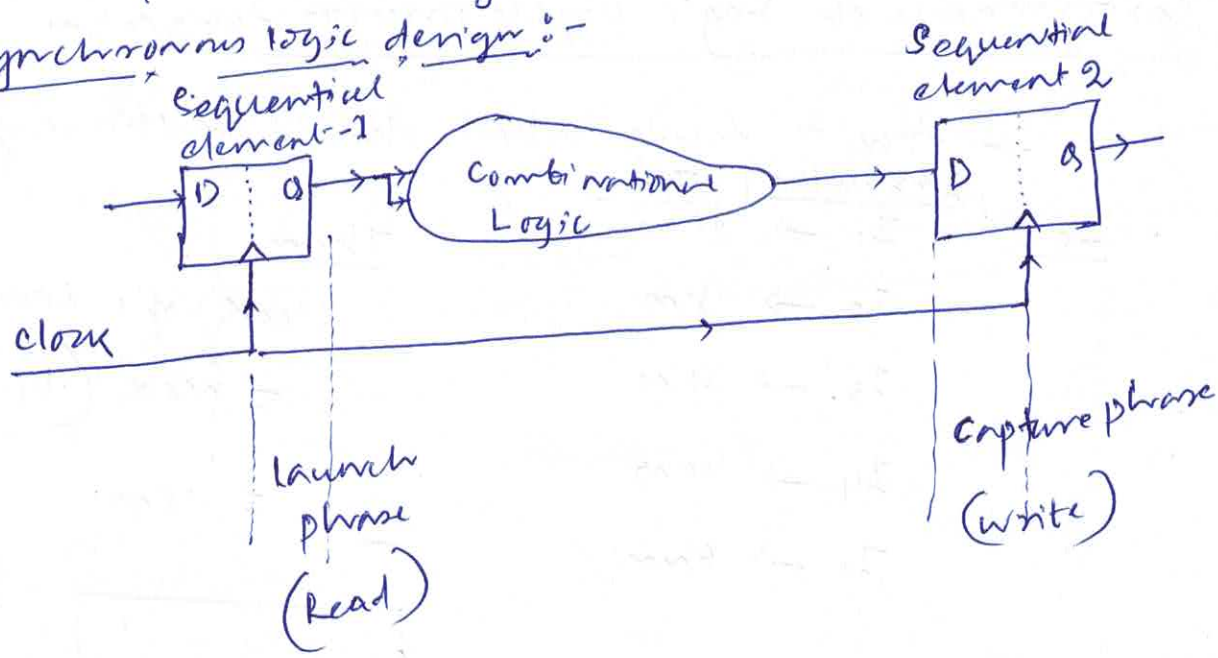
Lecture - 13 (23-10-2019)

Pipeline & Super scalar Architecture

- Single cycle design.
- Multi cycle design
- pipeline design
- super scalar (Multiple issue Architecture)

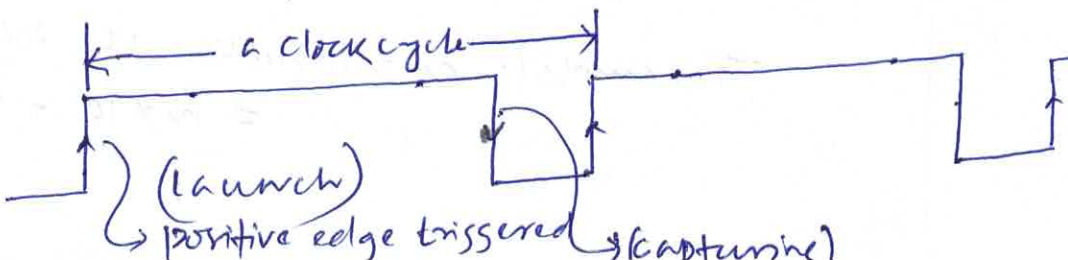
- The concept of ~~time~~ clock cycle arises

The synchronous logic design :-



- Launch and capture must be performed in distinct clock edge.
- If the launch is being performed in +ve edge the capture must be done in negative edge (why)
- What will happen if above is not done?

Analysis :-



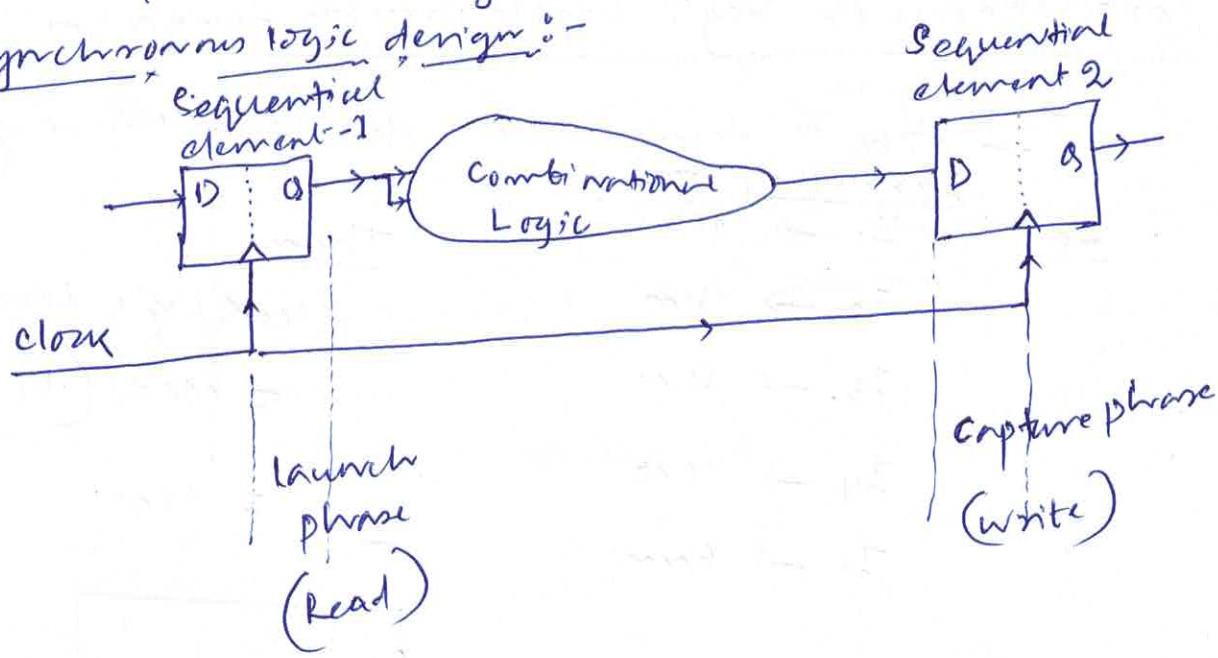
Lecture - 13 (23-10-2019)

Pipeline & Super scalar Architecture

- Single cycle design.
- Multi cycle design
- pipeline design
- super scalar (Multiple issue Architecture)

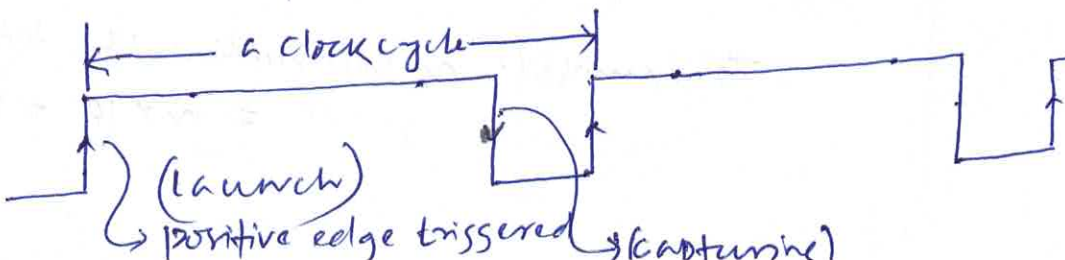
- The concept of ~~time~~ clock cycle arises

The synchronous logic design :-



- Launch and capture must be performed in distinct clock edge.
- If the launch is being performed in +ve edge the capture must be done in negative edge (why)
- What will happen if above is not done?

Analysis :-



(2)

Single cycle design -

- Data path for every instruction is single cycle
  - from fetch to write back everything takes place in a cycle.
  - The paths such as: memory to register, instruction memory to data memory and data memory to data memory are of single cycle type.

Performance of Single cycle processor design :-

- How to decide what should be the cycle length

Let-

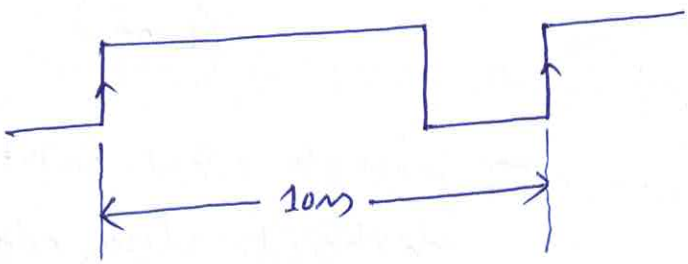
Instruction	Time
I_1	10 ns
I_2	8 ns
I_3	4 ns
I_4	10 ns
I_5	6 ns
⋮	⋮
I_n	5 ns

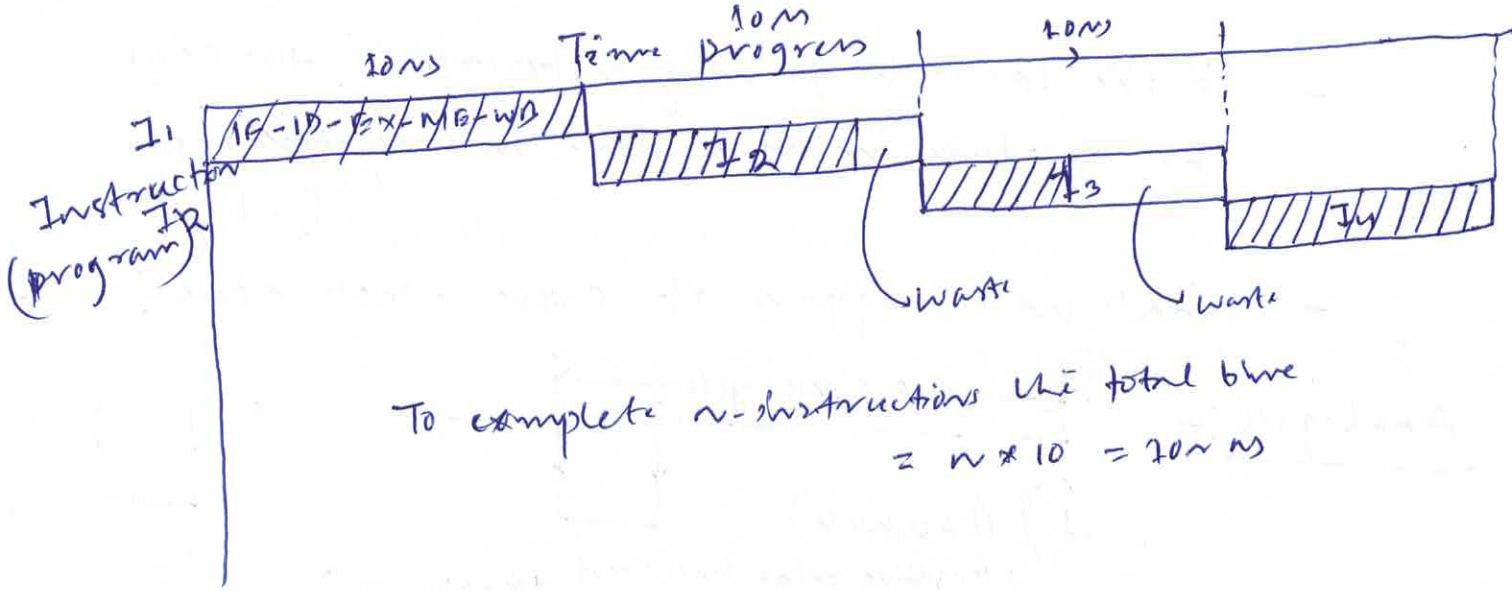
Then

Clock cycle length

$= \max(t_1, t_2 \cdots t_n)$

$= 10$ ns





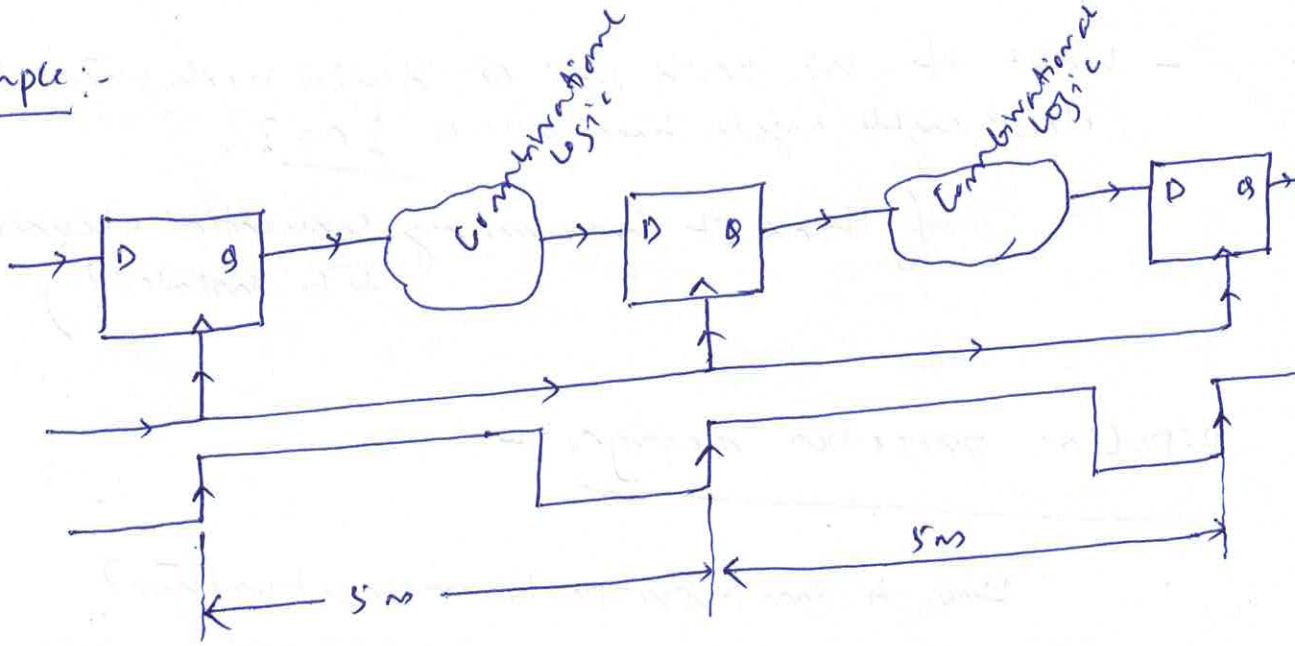
To complete n-instructions the total time

$= n * 10 = 10n$ ns

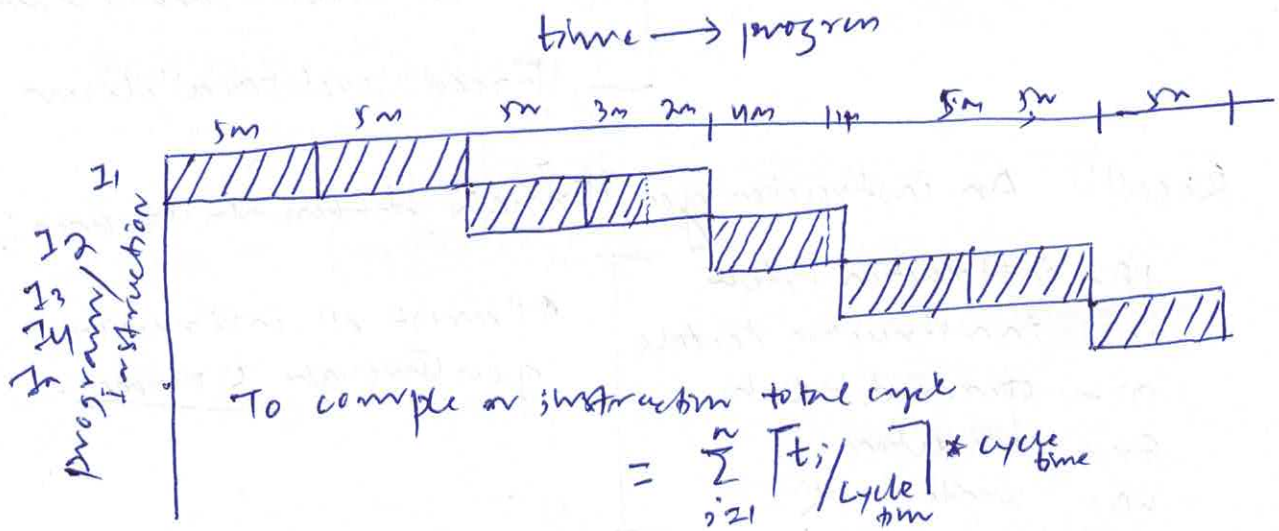
Multi cycle design :-

- How to avoid the waste time in single cycle design?
- Re-design the data path in such a way that the cycle time would be as optimal as possible.
- Break the combinational path by inserting additional sequential elements.

Example :-



Performance Analysis :-



To compute an instruction total cycle

$$= \sum_{i=1}^{n} \lceil t_i / \text{cycle time} \rceil * \text{cycle time}$$

Performance comparison:-

$$\text{Speedup} = \frac{\text{execution time in old processor}}{\text{execution time in new processor}}$$

$$= \frac{\text{time in single cycle}}{\text{time in multicycle}}$$

- What if the path will be divided such that the clock ~~cycle~~ cycle time will be 1 ns?

  (think of how many sequential elements to be inserted?)

Pipeline processor design:-

How to improve performance further?

- The concept of parallelism!
  - Instruction level parallelism
  - Data level parallelism
  - Thread level parallelism.

Recall: An instruction goes through different phases!

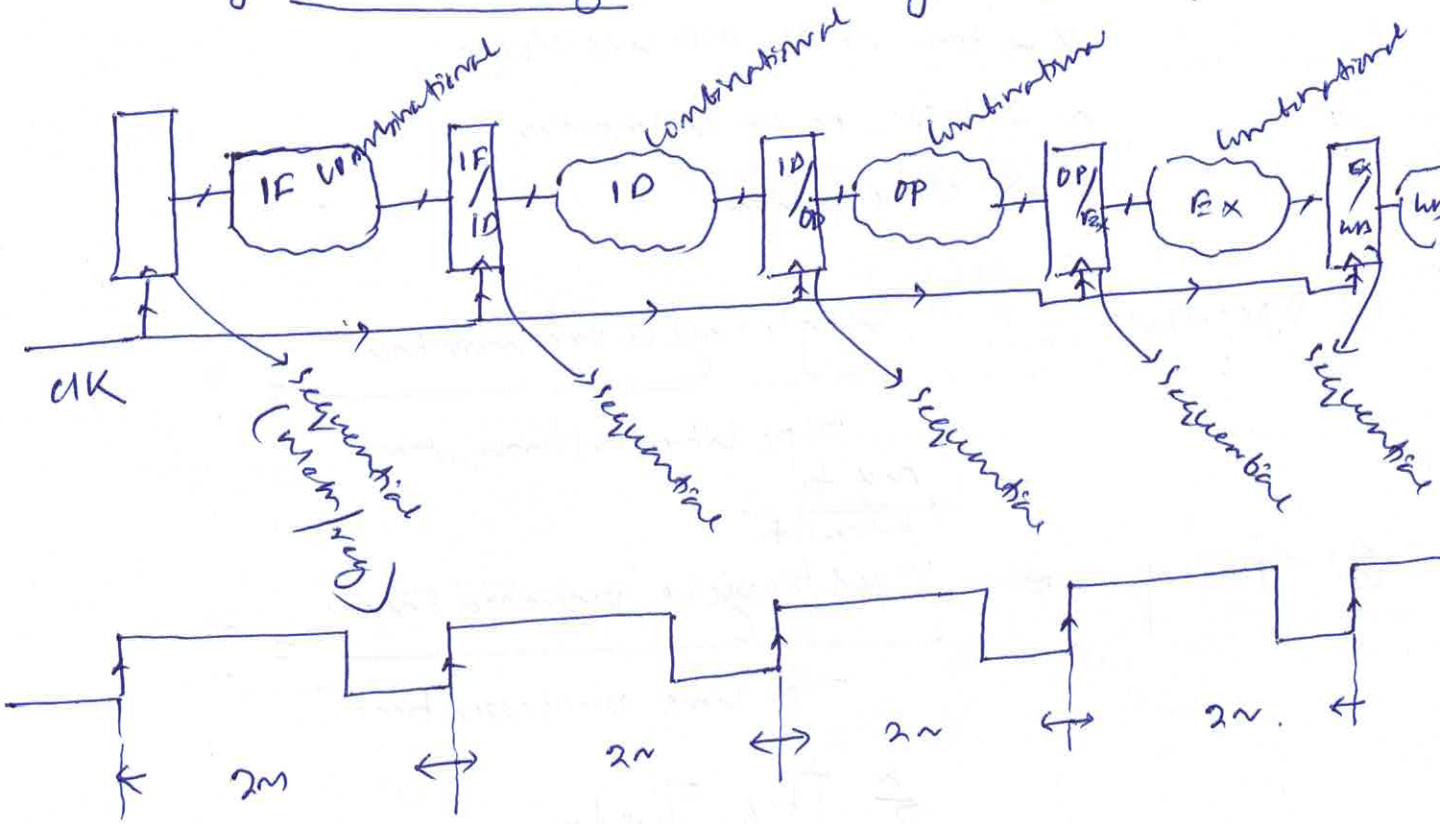
- IF - instruction fetch
- ID - instruction decode
- OP - operand fetch
- Ex - execution
- WB - write back

Almost all instruction goes through 5 phases.

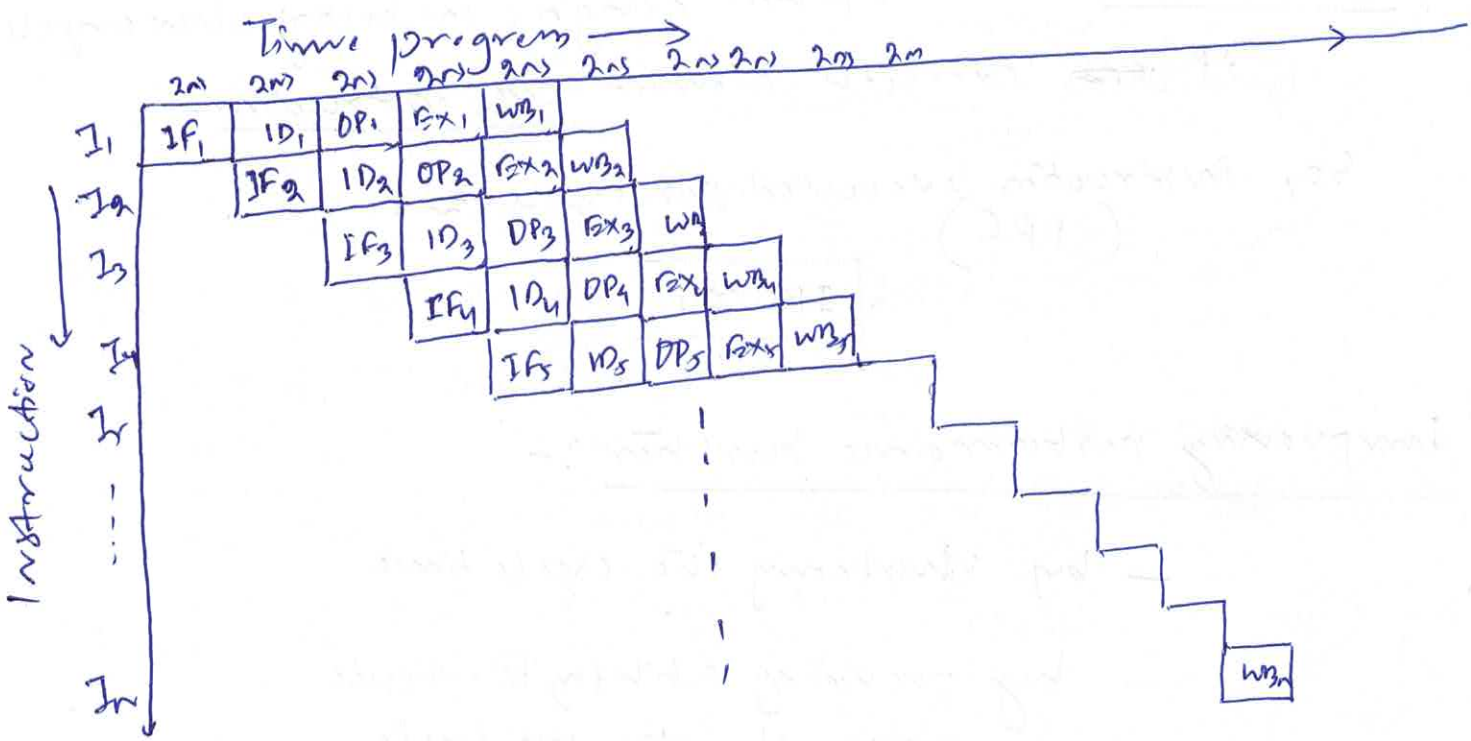
Solution – by executing multiple instruction simultaneously

– How ?

By clean partitioning between every two phases !



Performance Analysis :-



(6)

Total execution time = $K \times t + (n-1) * t$

$= (K+n-1) * t$

$K \rightarrow$ total no. of pipeline stages

$n \rightarrow$ total no. of instruction

$t \rightarrow$ clock period

① Speedup = $\frac{\text{Single cycle processor time}}{\text{Pipeline processor time}} = ?$

$= \frac{n * t_s}{(K+n-1) t_p}$

② Speedup = $\frac{\text{Multicycle processor time}}{\text{Pipeline processor time}} = ?$

$= \frac{\sum_{i=1}^{n} \lceil t_i / t_m \rceil * t_m}{(K+n-1) t_p}$

Observation :- In pipeline design, in every clock cycle it executes almost one instruction.

So, instruction executed per cycle (IPC) $\simeq 1$

$$\boxed{IPC \simeq 1}$$

Improving performance further :-

- by shortening the cycle time
- by providing ability to execute more instructions per cycle